初始化和赋值的区别

类作用域，构造函数，this指针，友元，可变成员，静态成员

类类型的操作包括：初始化，赋值，复制，撤销，分别对应着构造函数，赋值操作符，复制构造函数以及析构函数来处理这些操作。复制构造函数主要用在初始化，函数传参，函数值返回（感觉这一块没有什么会考的知识点，反正感觉有种学的很浅的感觉）。复制构造函数感觉需要考虑的主要是什么时候需要复制，什么时候禁止复制。（需要复制的时候就是类的成员含有指针，或者有成员表示在构造函数中分配的其他资源（这句话什么意思））（位拷贝，值拷贝，浅拷贝，深拷贝，一个不懂）

具体类和抽象类（设计虚函数和纯虚函数），虚函数表的原理，虚函数的实现原理

派生类和基类：虚函数和函数重定义相关，非虚函数直接继承，

还有一个重要的概念是动态绑定，动态绑定是指函数申明中使用的是基类的引用或者基类的指针，但是可以将基类和派生类赋值给他。在对应函数调用的时候，对于虚函数而言，调用的哪个版本取决于具体的实参。（强调，动态绑定只有通过基类类型的引用或指针来进行）默认实参不具有动态绑定性。

下面这句话说的很好：通过基类的引用或指针调用虚函数时，发生动态绑定。引用（或指针）即可以指向基类的对象，也可以指向派生类的对象，这一事实是动态绑定的关键。用引用或指针调用的虚函数在运行时确定，被调用的函数是引用或指针所指对象的实际类型所定义的。

派生类的成员包括继承基类的成员以及自己定义的成员。（派生类重定义基类的非虚函数，当使用基类指针指向派生类对象时，仍然将会调用基类版本的函数）

派生类和基类复杂的一个地方就在于基类类型的指针和引用可以指向派生类对象。（但是通过基类指针也只能够访问派生类对象的基类部分）

无论派生类列表中是什么访问标号，所有继承base的类对Base中的成员具有相同的访问。派生访问标号将控制派生类的用户对从base继承而来的成员的访问。（派生类对于基类成员的访问权限与访问标号无关）

派生类对象给基类对象赋值时，私有成员的赋值应该怎么处理？可以看做派生类对象时没有基类私有对象的（实际上有，但是派生类对象的私有对象有什么用呢？）派生类可能要访问基类已经写好的函数，而这个函数又要用到基类的私有成员，如果派生类不继承这个私有成员会怎么样？哦，我懂了！基类定义了一个私有成员a，然后写了一个公有函数输出这个a，派生类继承这个函数以后需要输出这个a，但是新建了派生类的对象，如果没有继承这个私有对象的话，那么就不存在这个a了。

接口继承和实现继承

声明而不定义一个派生类，应该怎么做？

覆盖虚函数机制，没有特别明白？

存储类的对象所用的存储空间分配：

对于这个问题，需要考虑基类和派生类，对于基类而言，需要考虑空的类，只包含非静态数据成员的类，包含静态数据成员的类，只包含成员函数的类（需要考虑特殊的虚函数）对于派生类而言，

静态成员变量：

可变成员变量：mutable

作用域操作符引用的类型成员

构造函数初始化列表：构造函数分两个阶段执行：（1）初始化阶段（2）普通的计算阶段。必须使用函数初始化列表的成员，包括const类型的成员以及引用类型的成员，没有默认构造函数的类类型的成员（这个没有看懂），初始化列表的顺序

默认构造函数和合成的构造函数不一样

构造函数，默认构造函数，合成的构造函数

Istream ostream这两个东西实在是一点都不懂。。。

内置类型的初始化

析构函数一点也不懂

几种排序算法的比较

类静态成员和全局对象 static数据成员，static成员函数

this指针

static变量：static全局变量，全局变量，静态局部变量，静态函数，（静态成员变量的派生有什么不一样的地方）**基类和其派生类共享该基类的静态成员变量内存！**

名字空间：全局名字空间，

Protected成员的使用方法：

C++编译，链接，执行各个部分分别做什么工作？？？

友元似乎没有什么重要的知识点

C++和继承相关的访问问题

子类型又是什么鬼？

大端小端老是记不住，草

I++ ++i还是记不住，草

虚函数的内存布局

成员方法又称为实例方法

静态方法又称为类方法

派生过程中构造函数和析构函数的调用顺序

C是D类中的子对象，执行派生类构造函数的顺序为：

1）调用基类构造函数；

2)调用自对象构造函数；

3）最后执行派生类构造函数本身。

编译时多态，运行时多态

C++哪些运算符不能重载？分量运算符 点号, 成员对象选择 点星号， 作用域解析 双冒号， 条件运算符        问号冒号。（为什么不能重载）

C++派生类中重新定义基类的变量？